

On the Use of the ADG Similarity in Case-Based Planning Systems*

Flavio Tonidandel² and Márcio Rillo^{1,2}

¹ Universidade de São Paulo - Escola Politécnica
Av. Luciano Gualberto 158, trav3
05508-900 - São Paulo - SP - Brazil

² Centro Universitário da FEI - UniFEI
Av. Humberto de A. Castelo Branco, 3972
09850-901 - São Bernardo do Campo - SP - Brazil

flaviot@fei.edu.br, rillo@lsi.usp.br

Abstract. *This paper is a further step in the investigation of the ADG similarity performance. The ADG similarity is a new similarity rule that was first used in the FAR-OFF case-based planning system, which presents good performance in solving planning problems. The ADG similarity is more accurate than old similarity metrics because it estimates the effort for adapting cases, but it takes more time to find similar ones. This paper analyses the real contribution of the ADG similarity in the FAR-OFF system performance and its advantages and disadvantages over the old similarity rules.*

Resumo. *Este artigo é uma investigação da performance da similaridade ADG, que é uma nova regra de similaridade que foi inicialmente usada no sistema FAR-OFF. A regra de similaridade ADG é mais precisa do que as antigas métricas de similaridade, isto porque ela estima o esforço de adaptação dos casos. Entretanto, ela gasta mais tempo para encontrar os casos similares. Este artigo analisa a real contribuição da similaridade ADG na performance do sistema FAR-OFF bem como suas vantagens e desvantagens com relação as antigas regras.*

1 Introduction

Case-Based Planning (CBP) systems are alternative approaches to generative planning. Most of them use old plans in a case-base - memory with plans as cases - for helping to solve problems, while generative planners construct plans from scratch.

The performance of the CBP systems is very dependent of the retrieval performance. The retrieval phase of a CBP system is responsible for finding a proper case to be adapted in order to solve a problem as fast as possible. The most important feature of the retrieval phase is the similarity rule, which is used to estimate the similarity of a case to the solution of a problem.

The similarity rules usually used by old CBP systems are not sufficiently accurate for finding suitable cases that are easily adapted to solve new planning problems. The ADG (*Action-Distance Based*) similarity [12] is a new and more accurate

* This work was supported by FAPESP under contract no. 98/15835-9.

similarity rule, which is based on the estimation of the number of possible actions that must be found to modify the case to a solution. Since the number of actions is a good estimation of the adaptation effort, the ADG similarity can find cases that are easier to be adapted, decreasing the entire time of a CBP system [12].

The ADG similarity was first used in the FAR-OFF system [13], and collaborates with the good performance of such system for solving many planning problems. However, to find accurate results, the ADG similarity takes more time to find proper cases than the similarity rules used in the past.

This paper investigates the collaboration of the ADG similarity on the FAR-OFF performance and if the time spent in the retrieval phase is really made up for by an easier adaptation of the case. This paper is a further step in the investigation of the ADG similarity performance. Tonidandel and Rillo [12] have shown that ADG similarity is more accurate than old similarity rules. This paper will investigate if this accuracy can improve the entire performance of a Case-Based Planning system.

Section 2 presents an overview of Case-Based Planning systems. Section 3 initiates the formalization of the planning components in Transaction Logic. Section 4 presents some details of the similarity rules considered in this paper. In section 5 some experiments are presented and discussed. Section 6 finally concludes the paper.

2 Case-Based Planning

Most planning systems are generative planners, which find solution from scratch. For the same problems, they always start the planning process from the beginning. The use of Case-Based Reasoning (CBR) techniques [6] can avoid a planner to repeat planning processes by considering some problems that were previously solved. The first application of CBR in planning was by the CHEF system [4], which received the denomination of Case-Based Planning (CBP) system.

A CBP system distinguishes from a generative planner by using a memory of previous executed plans¹. The CBP system, instead of starting a planning process from the beginning, it first looks for a similar solved plan in order to solve the new problem. Each previous plan is stored in a memory as cases with some additional features that specify the situation where the plan was applied. These features are used to indicate how much similar is a case to the solution of a new problem.

Usually, a similar case is retrieved by a retrieval phase, which uses a similarity rule to rank cases from the most to the least similar one. After the choice of the most similar case, a CBP system starts the adaptation process. In most of CBP systems, the adaptation phase is composed of a generative planner that permits the modification of a previous plan in order to find a new solution. This new solution, which is a plan, can be stored for future uses. These three processes – storing, retrieving and adapting – completes the cycle of a common CBP system.

Although theoretically the complexity of both is *NP-hard* [7], in practice CBP systems can outperform generative planning systems, because they avoid repeating the planning process for similar problems.

¹ There are CBP systems that do not consider cases as old plans, like the Prodigy/Analogy system. They use derivational analogy where cases are traces of old planning solutions. However, in this paper, we will focus only on cases as old plans

However, if the retrieval phase of a CBP system is not efficient, it hardly outperforms the new generative heuristic-search planning system, like the HSP [2] and the FF system [5]. The generative heuristic-search planners are much faster than the old generative planning systems.

One important feature that can increase the system performance is accuracy of the similarity metric, which can be accurate enough to anticipate the adaptation effort.

The most used domain independent similarity rules are based on the number of common features between the current problem and the stored case. These common features are obtained by the intersections between the current state and the initial state of the stored case and between the goal state and the final state of the stored case. If the total number, or normalized number, of the common features between a stored case and the current problem is higher than a specific limit, the stored case can be considered similar. In this approach the Footprint similarity rule [14] and the SNN (*Standard Nearest-Neighbor*) similarity rule [6] can be found. Most of state-space CBP systems use this kind of similarity rules.

However, this approach is not a suitable measure to improve the adaptation phase efficiency because it is not a good measure of the real adaptation effort. The FAR-OFF system [13] is the first state-space case-based planner that uses the ADG similarity rule [12] that anticipates the adaptation effort. However, the ADG similarity takes more time to determine similar cases, increasing the entire retrieval time. Therefore, it is important to investigate if the time spent by the retrieval phase can be made up for by the adaptation phase. The adaptation effort is supposed to be minimized because the ADG similarity considers similar cases those that require less adaptation effort.

This paper investigates the performance of the ADG similarity in comparison of old similarity rules in retrieval and adaptation times.

3 The Planning Components

Each component of a planning system must be defined. For that, this paper will consider the Transaction Logic (TR) [3], that is an extension of the first-order logic by the introduction of the serial conjunction operator (\otimes), e.g., $\alpha \otimes \beta$ means "first execute α , and then execute β ".

The TR is a suitable logic to describe actions and plans for planning systems [8, 10, 11]. Considering L a language defined in serial-Horn version of TR, the components of a planning system can be defined as:

Definition 1 (State): *The state D is a finite set of first-logic predicates and it is represented in TR as a database state. Each $d \in D$ is called fact.*

Definition 2 (Strips-like Actions): *Considering $A \subseteq L$ as a set of action definitions, each α , $\alpha \in A$, has the following structure: $\alpha \leftarrow pre(\alpha) \otimes delete(\alpha) \otimes add(\alpha)$, where*

- *$pre(\alpha)$ is a TR formula that is composed of $qry(_)$ predicates that represent the preconditions of the action*
- *$delete(\alpha)$ is the delete list of the action α . It represents the facts that were true before and will not be true after the action execution.*
- *$add(\alpha)$ is the add list of the action α . It represents the facts that were not true before and will be true after the action execution.*

The result of an action execution is an updated state D' from D after the deletion and insertion of the delete and add lists respectively. Any action α can be executed from a state D if $pre(\alpha) \subseteq D$. With actions and states, the following definitions are possible:

Definition 3: (Plan) A plan $\delta = \alpha_1 \otimes \dots \otimes \alpha_n$ is a TR formula, where $\alpha_i \in A$; $1 \leq i \leq n$.

Definition 4: (Goal) A goal Df is a TR formula and it is a set of queries that represents the desirable final state.

The purpose of a planner is to find a sequence of actions that transforms the initial state (current state) into a state where Df is satisfied. This plan can be stored as a case for future uses. A case is a plan connected with initial and final states features:

Definition 5: (Case) A case η is a TR rule: $\eta = Wi \otimes \alpha_1 \otimes \dots \otimes \alpha_n \otimes Wf$, where:

- $\alpha_i \in A$; $1 \leq i \leq n$, a plan defined by the planner that satisfies a proposed goal.
- Wi is a set of queries in TR that represents the precondition of the case.
- Wf is a set of queries in TR that represents the pos-condition of the case

Intuitively, Wi is a set of those facts that must be in the initial state and represents the pre-condition of the plan. It can be obtained by analyzing the actions of the plan and the facts that will be deleted from the initial state by the plan.

With respect to Wf , it is a set of those facts that are inserted by the plan and will be presented in the final state after the plan execution. The Wi and the Wf are the core part of the ADG similarity.

4 The Similarity Rules

There are many similarity rules that can be used in Case-Based Planning system. Most of them are domain independent and they are based on the differences between states, as highlighted in section 2.

This paper focuses on two old similarity rules. One is the traditional SNN (*Standard Nearest-Neighbor*) and another is the Footprint Similarity of the Prodigy/Analogy system [14]. These two similarities will be used in the time comparison against the ADG similarity metric [12]. In the following, a brief description of each similarity considered in this paper is presented.

4.1. The SNN Similarity

The Standard Nearest-Neighbor (SNN) is an old and often used similarity metric. It was introduced by Kolodner [6] as a rule of similarity for Case-Based Reasoning systems. It is based on a formula that considers the difference between two set of facts.

The general formula considers the computation of the differences between two features of a same slot. For each slot, a suitable weight is used to differentiate the importance of one slot from another. In Case-based planning, there are no slots and features, but only sets of facts, which describe states or part of them. Since each fact has no value, the SNN rule can be described in a reduced form as:

$$SNN = |Si \cap Sc| \quad (1)$$

where Si is the set of facts from the new problem and Sc is a set of facts from a case.

There are two important features that can be aggregated to the formula 1 that can increase the accuracy of the SNN value.

The first is to consider some weights for facts, or combination of facts, in formula 1. These weights are domain dependent features and they would be a part of the domain knowledge that must be provided by the user to the system. Since it is not a feasible feature for domain independent systems, it will not be considered in this paper.

The second feature that can be considered is the calculation of a normalized value instead of only the number of facts that two sets interchange. It can be done by dividing the formula 1 by $|Scl$. Since the number of facts in a state (set) can vary from problem to problem, the normalized formula is more accurate than formula 1.

In this paper, the SNN rule will be defined and used as a normalized formula. The intersection is between the initial state and Wi of the case, and between the goal and the Wf of the case. The normalized value of the SNN is the sum of them:

$$SNN = \frac{|D0 \cap Wi|}{|Wi|} + \frac{|Df \cap Wf|}{|Wf|} \quad (2)$$

This SNN rule value is improved by the use of the Wi and The Wf features. Because of that, it is more accurate than the footprint similarity of the Prodigy/Analogy system in some situations.

4.2. The Prodigy Footprint Similarity

Veloso [14] introduces a more detailed similarity rule that footprints the initial state with some relevant facts. The process of footprint is similar to the process that determines the Wi of a case.

The main idea is to footprint some features of the initial state. It is made by observing which goal facts of the new problem match the goal facts of a case. Thus, for each matched goal fact, their correspondent facts, or footprint facts, are determined in the initial state.

The determination of footprint facts, in a general view, is made by goal regression by using the actions in the plan of a stored case. First, the action that inserts a specific matched goal fact is selected. Its precondition becomes new matched facts for the actions before in the plan. The process goes until the facts are in the initial state. These facts are the footprint facts. The process is made for each matched goal fact.

Any one can note that the footprinted initial state is a subset of the Wi or equal to the Wi at most. The process to determine Wi is similar to the footprint process, but considering all goal facts, not only the matched facts. The Footprint similarity is considered as:

$$F\text{-Print} = |D0 \cap FSi| + |Df \cap Wf| \quad (3)$$

where FSi is the foot-printed initial state for the set of matched goals ($Df \cap Wf$).

The original Footprint similarity does not consider normalized values. Then, in this paper it does not either.

4.3. The ADG Similarity

The ADG (*Action-Distance Guided*) similarity [12] is a similarity metric that has as a purpose to estimate the adaptation effort. This estimation is based on the number of actions that is necessary to transform a case to a solution of a problem.

The ADG similarity is a process that uses the Wi and the Wf of the stored cases and calculates two estimate values of the distance between states based on a number of actions. The first value is called *initial similarity value* (δ_i), and it estimates the distance between the current initial state, denoted by D_0 , and the initial state features of the case, denoted by Wi . The second value is called *goal similarity value* (δ_G) and it is a distance estimate between the desirable final state (Df) of the goal (def. 4) and the final state features (Wf) of the case.

The process of estimating the distance between two states is obtained by using the heuristic of the FF planner [5]. This heuristic was the main responsible for the excellent performance of the FF system in the AIPS'00 planning competition [1].

The first step for determining the FF's heuristic is to create a graph from the initial state by ignoring the delete list of actions. As defined in [5], this graph is constituted by layers that comprise alternative facts and actions. The first fact layer is the initial state (D_0). The first action layer contains all actions whose preconditions are satisfied in D_0 . Then, the add lists of these actions are inserted in the next fact layer together with all facts from the previous fact layer, which leads to the next action layer, and so on. The process keeps going on until it finds a relaxed fixpoint, i.e., when there are no more fact layers that are different from previous fact layers.

Some useful information can be determined from the relaxed fixpoint process. Following [5], they are:

- Definition 6:**
1. $level(d) := \min \{i \mid d \in F_i, \text{ where } F_i \text{ is the } i^{th} \text{ layer of facts} \}$
 2. $level(\alpha) := \min \{i \mid \alpha \in O_i, \text{ where } O_i \text{ is the } i^{th} \text{ layer of actions} \}$

The definition 6 provides the order number of the layer where each fact or action appears first. It means that each fact, or action, is a membership of the layer that it first appeared.

With the relaxed graph, it is possible to find a relaxed solution for any state that can be reached from D_0 . This is done by selecting the action responsible to insert each goal fact. This selection is performed from the last layer of the graph to the first layer. The process of selection continues by initialized each fact of the precondition of the selected action as a new goal in its correspondent layer. The process stops when all unsatisfied goals are in the first layer. This process, extracted from [5], is shown in Figure 1a, where the variable h is used to count the number of selected actions.

The *initial similarity value* (δ_i) is directly obtained by the determination of the relaxed solution from D_0 to Wi . The *Initial Similarity Value* is the result h of the function **relaxed_initial_length** (Wi) after *setting all marks of all facts as false*.

In order to calculate the second value δ_G , it is necessary to force the solution trace from D_0 to consider the actions in the case. To do this, it is necessary to maintain and use the marks changed by **relaxed_initial_length**(Wi), to set all marks of all facts in Wi as false and the marks of each fact in Wf as true in their correspondent layer.

<pre> relaxed_initial_length(G) clear all Gi for i:= 1 ... max do Gi := {g ∈ G level(g) = i}; endfor h:=0; for i:= max ... 1 do for all g ∈ Gi, g False at time i do select α_{level=i-1}; g ∈ add(α); h:=h+1; for all d_{level ≠ 0} ∈ pre(α), d not True at time i-1 do G_{level(d)} := G_{level(d)} ∪ {d}; endfor for all d ∈ add(α) do mark d as True at time i-1 and i; endfor endfor endfor endfor return h; </pre>	<pre> relaxed_final_length(Wi, Wf, Df) G:=Df; for each d ∈ Wi do mark d as False at all levels; endfor for each d ∈ Wf do mark True at level(d) mark True at level(d)-1 endfor h' := relaxed_initial_length(G); return h'; </pre>
(a)	(b)

Figure 1. Both algorithms used in the ADG similarity estimation. (a) The algorithm that computes the relaxed solution from a relaxed graph, where G is the target-state. It is extracted from [5]. (b) The algorithm that extracts the distance between Wf and Df , considering the marks created by the *relaxed_initial_length(Wi)*.

This is done by the *relaxed_final_length(Wi, Wf, Df)* (Fig. 1b), which finds the second part of the similarity metric: $\delta_G = h'$. Finally, the ADG similarity value can be determined by $\delta_i + \delta_G$. This formula and the algorithms will be considered the ADG similarity for the rest of the paper.

5 The Experiment

In the experiment, the SNN, the ADG and the Prodigy Footprint similarities will be considered. The tests will consider the time for adapting and retrieving cases.

There will be two time tests. The first one will analyze the time to retrieve cases from case-base by each similarity. The second one will consider the accuracy of each similarity rule by analyzing the time spent to adapt the retrieved cases.

For both tests, two domains will be considered: The Blocks World and the Logistic domains. For each domain, a set of planning problems will have to be solved by the FAR-OFF system [13].

The FAR-OFF system is a case-based planning that has a FF-based generative planner to adapt cases. It also uses a Footprint-Based Retrieval [9] to decrease the number of cases to be considered by the retrieval phase.

For solving each problem, a random generated case-base is considered. They are created by a General Case-base Seeding system. The experiments will consider the same case-bases used in the FAR-OFF system paper [13]. Some details of the case-base seeding system can be also obtained in [13].

5.1. The Retrieval Time

The problems and the case-bases are those used in [1, 13]. For blocks world domain they range from problem-4-0 with 4 blocks to problem-28-1 which has 28 blocks. In Logistic domain, the problems range from problem-4-0 to problem 15-1. More details about these domains can be obtained in [1].

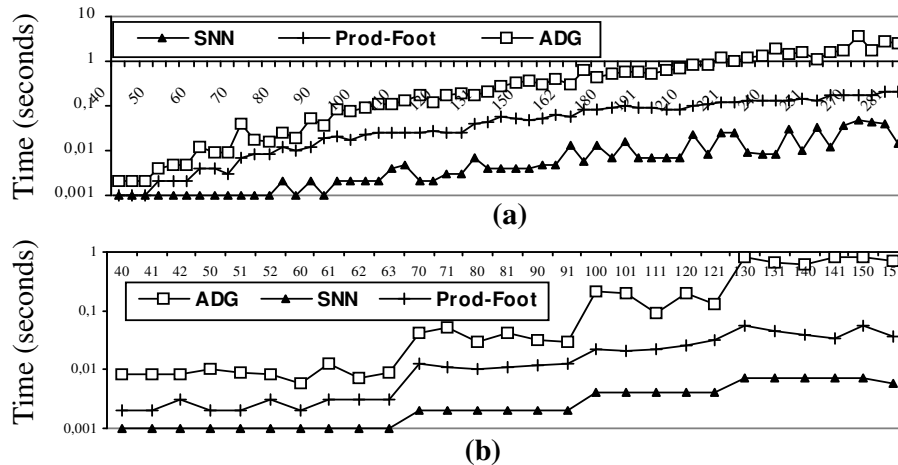


Figure 2 – Time for retrieving cases by using the FAR-OFF system with different similarity rules in Blocks World Domain (a) and in Logistic domain (b). The time shown in Y-axis is in seconds and in logarithmic scale.

The case-bases used in the tests range from 50 cases for 4 blocks to 2450 cases for 28 blocks problems. In logistic domain, it ranges from 200 cases to 1400 cases.

This test shows the time spent for each similarity to find and retrieve similar cases by using the FAR-OFF retriever engine [13] in both domains.

The results are shown in figure 2. It is possible to observe that the retrieval time of the ADG similarity degrades rapidly, while the time to retrieve cases with SNN and FootPrint similarities are smaller.

One important feature that is responsible for the ADG similarity performance are the case-bases used in the test. They were randomly created by a Case-Based Seeding that does not necessarily create good samples of cases. It increases the number of redundant cases which do not permit the suitable creation of the Footprint set of cases by the Footprint-Based Retrieval [9].

In consequence, the Footprint-based Retrieval, that should consider only a subset of cases, considers almost all cases in the case-base.

One way to overcome this problem is to refine the random case-bases or even find another method to reduce the space of cases or change the Footprint-based Retrieval.

If the number of cases is reduced, the time to retrieve cases with the ADG similarity will decrease much more than the SNN and FootPrint times. Since that, the ADG similarity time will be almost equal to Footprint time and close to the SNN time in most situations.

5.2. The Similarities Accuracy

After the retrieval phase, a CBP system must adapt the retrieved case or cases. It is done by the adaptation phase. The accuracy of the similarity rule will affect drastically the adaptation phase, because if a retrieved case is not similar to the problem, the adaptation phase will spend much more time to find a proper solution..

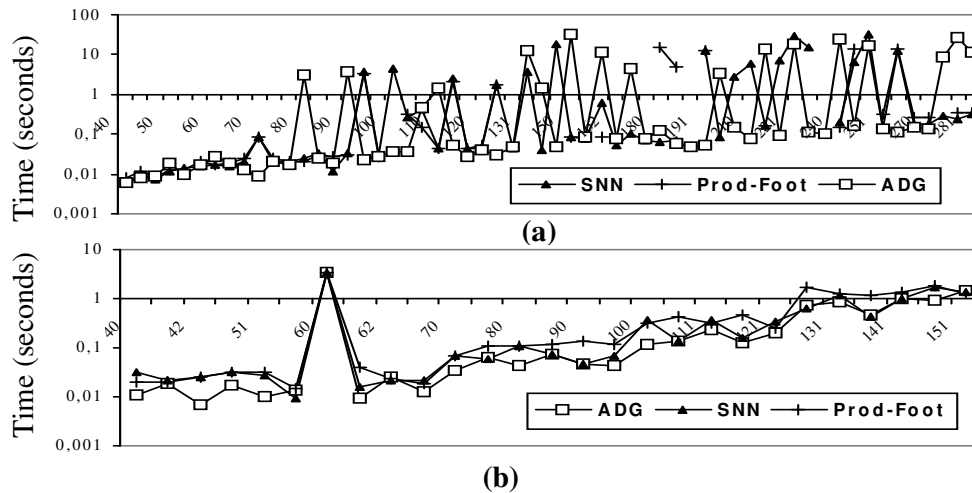


Figure 3 – Time for adapting cases by using the FAR-OFF system with different similarity rules in Blocks World Domain (a) and in Logistic domain (b). The time shown in Y-axis is in seconds and in logarithmic scale.

This test will analyze the time spent to adapt a retrieved case for finding a solution. There are 5 retrieved cases for each problem at most. The retrieved cases and the problems are those used in the previous test.

The figure 3 presents the performance of each similarity in both domains. In most of situations (problems) in each domain, the ADG similarity presents the smaller time for the adaptation of the cases. It means that the ADG similarity is more accurate than the other two considered in the tests.

The peaks – sharp points in the graph - show that the problem was not solved by the first retrieved case, but by one of the other ordered 4 retrieved cases. The peaks incorporate the time wasted to adapt a case without solution.

For 52% of the Blocks World problems, the ADG similarity presents the fastest results. The SNN presents the fastest results for 36% and the Footprint for only 12%. In Logistic domain the ADG is even better. It solves faster 74% of the problems than the others. The SNN is the fastest way for 22% and the Footprint for 4%. It is important to notice that the time scale is a logarithm scale, which means that little differences in the graph can correspond to large numerical differences.

Another important feature is that the system with the ADG similarity solves 100% of the problems in Blocks World domain, while with the SNN it solves 90% and with the FootPrint it solves 50% of the total problems. It shows that the ADG is the main responsible by the FAR-OFF system for solving 100% of the problems.

However, the accuracy of the ADG is not enough to guarantee that most of the problems can be solved faster than by other similarity rules. Considering the total time, adaptation plus retrieval times, the ADG similarity is the fastest way for 41% of the problems in Logistic domain, while the SNN is the fastest for 55% and the FootPrint is only for 4%. In Blocks World, the ADG is the fastest way for 33% of the total problems, the SNN is the fastest for 64% and FootPrint is only for 3%.

The accuracy can make up for, in few cases, the time spent in retrieval phase. However, in most situations the retrieval time really degrades the overall performance

of the FAR-OFF system. It shows that, as highlighted in last section, a new and more accurate method to reduce the space of cases in the case-base is strongly necessary.

6 Conclusion

The ADG similarity is more accurate than old similarity metrics, which were represented by the SNN and the F-Print similarities. Its accuracy permits a case-based planning system, like the FAR-OFF system, to solve most of problems in planning domains. In the tests, the FAR-OFF system solved 100% of the problems considered in the experiments using the ADG similarity.

However, although the ADG similarity presents accurate results, it degrades the time of the retrieval phase. This time degradation can affect the performance of a case-based planning system. Therefore, a method that reduces the set of cases considered in the retrieval search engine must be investigated in the future.

References

- [1]. Bacchus, F. AIPS-2000 Planning Competition Results. Available in: <http://www.cs.toronto.edu/aips2000/>.
- [2]. Bonet, B; Geffner, H. 2001. Planning as Heuristic Search. *Artificial Intelligence*. 129: 5-33.
- [3]. Bonner, A.J.; Kifer, M. 1995. Transaction logic programming. Technical Report, CSRI-323, Department of Computer Science, University of Toronto.
- [4]. Hammond, K. 1989. *Case-Based Planning: Viewing Planning as a Memory Task*. Academic Press.
- [5]. Hoffmann, J.; Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research*. 14: 253 – 302.
- [6]. Kolodner, J. 1993. *Case-Based Reasoning*. Morgan Kaufmann.
- [7]. Nebel, B. ; Koehler, J. 1995. Plan reuse versus plan generation: A theoretical and empirical analysis. *Artificial Intelligence - Special Issue on Planning and Scheduling*. 76: 427-454.
- [8]. Santos, M.; Rillo, M. 1997. Approaching the *Plans are Programs* Paradigm using Transaction Logic. In: *Steel, S., Alami, R (Eds) Proceedings of 4th European Conference on Planning – ECP'97*. Lecture Notes in Artificial Intelligence, vol. 1348. 377-389. Springer-Verlag.
- [9]. Smyth, B.; McKenna, E. 1999. Footprint-Based Retrieval. In: *Althouff, K., Bergmann, R., Branting, K. (Eds.) Proceedings of the 3rd International Conference in Case-Based Reasoning. ICCBR'99*. Lecture Notes in Artificial Intelligence, Vol 1650. 343-357. Springer-Verlag. .
- [10]. Tonidandel, F.; Rillo, M. 1998. Case-Based Planning in Transaction Logic Framework. In: *Proceedings of Workshop on Intelligent Manufacturing Systems (IMS'98)*. 281-286. Elsevier Science.
- [11]. Tonidandel, F.; Rillo, M. 2000. Handling Cases and the Coverage in a Limited Quantity of Memory for Case-Based Planning Systems. In: *Sichman, J., Monard, C. (Eds). Proceedings of IBERAMIA/SBIA 2000*. Lecture Notes in Artificial Intelligence, Vol 1952. 23-32. Springer-Verlag.
- [12]. Tonidandel, F.; Rillo, M. 2001. An Accurate Adaptation-Guided Similarity Metric for Case-Based Planning In: *Aha, D., Watson, I. (Eds.) Proceedings of 4th International Conference on Case-Based Reasoning (ICCBR-2001)*. Lecture Notes in Artificial Intelligence. vol 2080. 531-545. Springer-Verlag.
- [13]. Tonidandel, F; Rillo, M. 2002. The FAR-OFF system: a Heuristic Search Case-Based Planning. In: *Proceedings of the 6th Conference on Artificial Intelligence Planning and Scheduling (AIPS'02)*. Toulouse. France.
- [14]. Veloso, M. 1994. Planning and Learning by Analogical Reasoning. *Lecture Notes in Artificial Intelligence*, 886. Springer-Verlag.